An Architecture for the 5G Control Plane based on SDN and Data Distribution Service

Alejandro Llorens-Carrodeguas
Universitat Politècnica de Catalunya
Barcelona, Spain
Email: alejandro.llorens@entel.upc.edu

Cristina Cervello-Pastor
Universitat Politècnica de Catalunya
Barcelona, Spain
Email: cristina@entel.upc.edu

Irian Leyva-Pupo
Universitat Politècnica de Catalunya
Barcelona, Spain
Email: irian.leyva@entel.upc.edu

Juan Manuel López-Soler
University of Granade
Granade, Spain
Email: juanma@ugr.es

Jorge Navarro-Ortiz
University of Granade
Granade, Spain
Email: jorgenavarro@ugr.es

***Abstract*—The tremendous growth of services and costumers' demands have rendered traditional networks inefficient. Telecommunication operators need a more flexible, scalable, faster and programmable architecture to offer users these new services. Software Defined Networking (SDN) has emerged as a natural solution to this situation as it enables network programmability. This article provides a review of the SDN architectures applied to fifth generation (5G) networks. In this work, the prime focus is a proposal of control plane for a 5G architecture with a hybrid hierarchical set of controllers. The architecture is based on a federation of multiple sub-network controllers, each managing only a section of the network, conveniently coordinated by a hierarchically-superior controller. The use of Data Distribution Service (DDS) as a standard of the Object Management Group (OMG) is explored to improve the performance of the proposed architecture. DDS is used taking into account empirical results which have demonstrated a significant improvement in the performance compared to other existing solutions that do not use DDS. We illustrate the flexibility of our approach by presenting some use cases describing how the different elements of this architecture works.**

***Index Terms*—Software Defined Networking, Control Plane, 5G, Area Controller, Global Controller, Data Distribution System.**

I. Introduction

There is no denying that the 5G architecture has become a hot topic for network operators. This might be due to the explosion of mobile devices along with the appearance of new types of applications and services such as augmented reality, virtual reality and so on. Therein, mobile network operators are feeling driven to upgrade their systems and invest in their infrastructure to satisfy customer demands. However, 5G does not only carry mobile traffic but also fixed and WiFi traffic and so on. 5G represents one step forward in comparison with previous generations. Operators want a simple architecture in which a variety of technologies are able to coexist, ensuring the satisfaction of customers' demands.

Software Defined Networks (SDN) entails the opportunity for new design principles on how networks should be developed, deployed and operated. The advantages of decoupled data plane and control plane deliver new means and methods to instantiate network functions and services, reducing expenses and boosting performance.

The initial design and implementation of OpenFlow assumed a single controller for the sake of simplicity. However, as the number and size of production networks deploying OpenFlow increases, the network feasibility decreases. For this reason, it is much better use distributed SDN controllers to large-scale networks.

Data Distribution Service (DDS) is a OMG's standard middleware for distributed real-time applications which it is based on Publisher/Subscriber paradigm. It simplifies the application development, deployment and maintenance and provides fast and predictable distribution of time-critical data over a variety of transport networks. Furthermore, DDS provides a flexible data distribution infrastructure for integrating data sources of all types. It delivers large amount of data with microsecond performance and granular Quality of Service (QoS) control using a distributed cache, referred to as data-space. DDS decouples in time and location the data producers and consumers [1].

In this work, we focus on proposing a new architecture for the 5G Control Plane using two-level SDN Controllers in which the top controllers are communicated by means of DDS. The following section contextualizes the addressed problem and provides related works. In Section III, we explain the applicability of using DDS in our 5G Control Plane architecture taking into account previous results integrating DDS with SDN Controllers. Section IV presents the proposed architecture for the 5G Control Plane and provides some insights about the controller components. Section V illustrates two use cases of our proposal. Finally, conclusions are given in Section VI.

II. Related Work

Several attempts to define a 5G architecture based on SDN are currently available in the literature.

In [2], the authors presented a two-level architecture of SDN Controllers. The bottom layer is formed by the area controllers, which are connected to physical switches and routers. The domain controllers are in the upper layer, which control the area controllers as devices, and synchronize the global abstracted network view through a distributed database. However, it is not efficient in terms of latency because the domain controllers depend on the area controllers to acquire network information.

A unified Control Plane for 5G is proposed in [3] made by three logical controllers: Device Controller, Edge Controller and Orchestration Controller. Those controllers run different applications to implement the 5G control plane. This allows 5G networks operators to dynamically instantiate logical architectures, implement network functions and services in the optimal location according to network requirements. The authors show a significantly reduction of latency even though it is far away from the 1 ms 5G latency requirement [4].

A programmable all-SDN 5G network architecture is introduced in [5]. This architecture eliminates special and expensive elements such as: Mobility Management Entity (MME), Serving Gateway (S-GW), Packet Gateway (P-GW) and Policy and Charging Rules Function (PCRF). This approach has user traffic forwarding based on IP flow entries instead of using GTP tunneling mechanism which improve the time for such dispatching process. In addition, the authors agree that a set of hierarchical controllers instead of a single centralized controller is necessary to deal with the delay constraints associated with various control functionalities of the mobile network.

There are many other papers, yet their contributions are generic, lacking focus and key details on the novel enabling technologies. In [6] a survey of different literatures related to this topic is made.

On the other hand, several attempts have been done to distribute SDN controllers and exchange network information among each other.

In [7], the authors propose a distributed event-based control plane for OpenFlow called HyperFlow, which allows network operators to deploy any number of controllers in their network. HyperFlow provides scalability while keeping network control logically centralized. Each controller publishes events related with the state of the system, while others controllers replay all the published events to reconstruct the state. HyperFlow use Publish/Subscribe paradigm to facilitate cross-controller communication by means of WheelFS, which is a distributed file system designed to offer flexible wide-area storage for distributed applications.

Similarly, in [8] it is proposed a distributed system which runs on a cluster of one or more physical servers, which may run multiple Onix instances. These instances are responsible for disseminating network state to other instances within the cluster. The network state is stored in a data structure called Network Information Base (NIB). The NIB is a graph of all network entities within a network topology. ONIX provides scalability and resilience by replicating and distributing the NIB between multiple running instances.

These approaches, despite their ability to distribute SDN control plane, impose a consistent network-wide view in all the controllers and thus generate large control traffic.

In [9] [2] the authors propose two types of controllers: the domain and the area controllers. Furthermore, they describe the modules that compose each controller and the relationship among them. They proposed a horizontal communication module which is responsible for synchronizing global abstract network information among the domain controllers. In order to do this, they use an scalable NoSQL database to store global host information, global switch information and global abstract topology information. The distribution of routing rules is realized through the Publish/Subscribe mechanism.

Phemius et al. [10] propose DISCO, a controller that manages its own network domain and communicates with other controllers to provide end-to-end network services. This communication is based on a lightweight and highly manageable east-west control channel. They implemented DISCO on top of the Floodlight OpenFlow controller and utilized AMQP to establish communication with others controllers. The *Messenger* and the *Agents* are two key elements in DISCO. The first one discovers neighboring controllers and maintains a distributed publish/subscribe communication channel; while the second one uses this channel to exchange network-wide information with other controllers.

There are others papers where their authors use both SDN and DDS to communicate controllers and switches. In [11] [12] [13], the authors propose DDS to exchange network state information related with QoS parameters between switches and controllers. Despite the fact that these papers use both paradigms in order to improve network behavior, their contributions are focused on vertical communication.

Our purpose is to federate SDN controllers by means of DDS, in this way they will exchange network information among each others. Thus, in case of failure of one controller the other one can assume the network control because they share the same view of the network.

III. Applicability of using DDS

In this section, we present further details on how implement and integrate the modules of Publisher and Subscriber with Opendaylight controllers. In addition, it is explained some obtained results using DDS to federate two Opendaylight controllers.

A virtual machine where software like Maven, Eclipse and a base distribution of OpenDaylight controller are installed by default is used. However, it is necessary install the RTI Connext DDS on the virtual machine in order to implement the design.

Maven was used to generate a compatible archetype with application structure of Opendaylight controllers. Once the project has been generated with Maven, it is necessary to install the *nddsjava.jar* library and modify the *pom.xml* file to add it this dependency. In this file are specified the dependencies (package of Maven and OpenDaylight repositories)

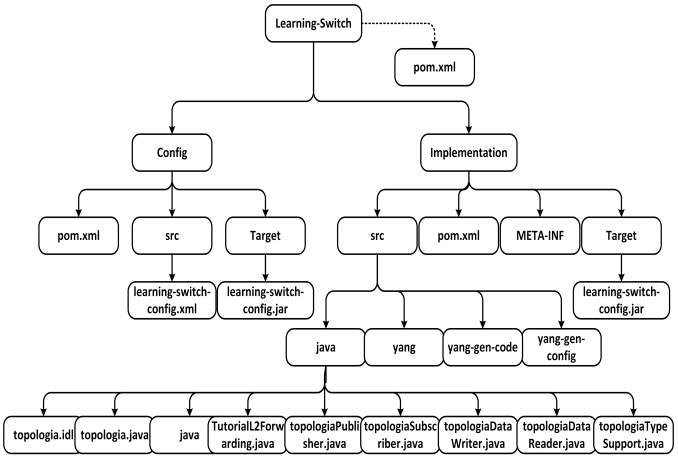

Fig. 1: Project structure.

that must be loaded by the application during compilation and execution times.

Two controllers were used for the evaluation and two projects were generated with Maven: one for the Publisher and the other for the Subscriber. Both projects have the same structure, the only difference is in the code of their classes. Fig. 1 shows the different classes and folders that compose these projects.

The performance of design is evaluated using an emulated SDN-based network such as Mininet. We run a tree topology with two remote controllers as shown in Fig. 2.

In this scenario, we have defined some cases to measure the recovery time needed in case of network failure. The results are shown in Fig. 3.

Case I: This is the simplest situation, as a single controller is used in the topology. In this case, when the controller fails, it has to reboot itself to come back to the initial state. The

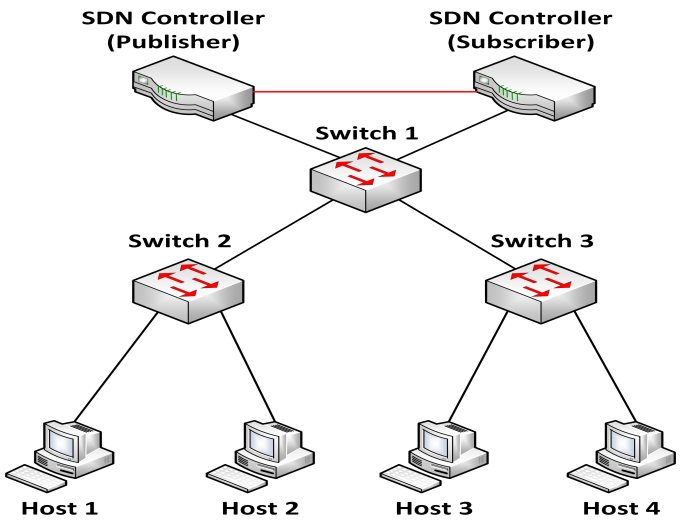

Fig. 2: Network topology.

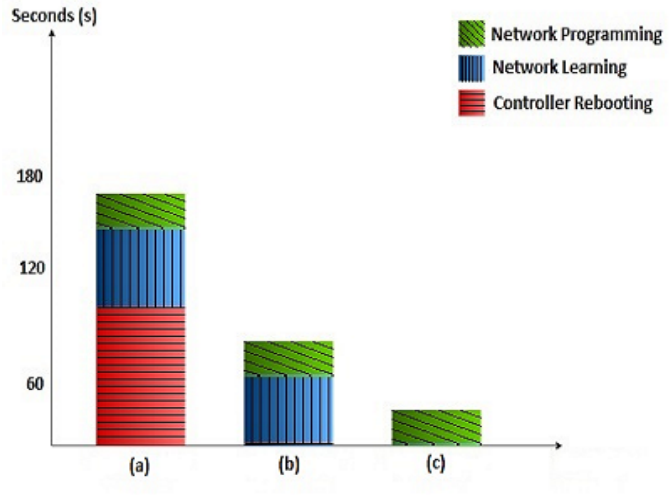

Fig. 3: Fail recovery time in different scenarios: a) one controller; b) two controllers without DDS; c) two controllers with DDS.

controller rebooting time in this case is about 100 s.

Case II: The second situation uses two controllers (one active and the other in stand-by) without DDS. By doing so, controllers do not share any information. When the active controller fails, the other one takes over the network control. This action will clearly impact service delay, as the controller needs to learn the network topology. Compared to the previous situation, there is no improvement in terms of network programming and network learning time in case of failure.

Case III: The last situation utilizes two federated controllers using DDS. In this case, both controllers are sharing network information. When the active controller fails, the stand-by controller takes over the network control. Unlike the previous case it does not need to learn the network topology because it was shared by the active controller by means of DDS. In this case, the reliability and resilience of the network has been improved.

DDS simplifies the development, deployment and maintenance of applications and provides fast and predictable distribution of time-critical data over a variety of transport networks. Furthermore, this application provides a flexible data distribution infrastructure to integrate data sources of all types.

DDS delivers large amount of data with microsecond performance and granular quality-of-service control as a result of its Dynamic Discovery characteristic for publishers and subscribers. This means that the application does not need to know or configure the communications endpoints, since they are automatically discovered. The communication is established using Topics, which are named data streams of the same data type. A topic named Topology was defined to share network information such as: Identifier, Node Id, Termination Point Id, Link Id, Source and Destination Node and Flows. By means of DataWriters and DataReaders, Publishers and

Subscribers can publish and subscribe DDS samples into and from the streams. More than one Topic can use the same user data type, but each Topic needs a unique name [1].

IV. 5G Control Plane Architecture Proposal

Despite the standardization efforts of 5G during the last years, just recently the first standard for 5G system architecture was defined by the 3GPP in its specification TS 23.501 [14].

The 5G 3GPP architecture is an evolution of the current 4G MBB, based on the concepts of control and user planes split, service base architecture and network slicing. This architecture consists mainly of the following network functions: Authentication Server Function (AUSF), Access and Mobility Management Function AMF), Network Slice Selection Function (SSF), Policy Control Function (PCF), Session Management Function(SMF), Unified Data Management UDM), User Plane Function (UPF), Application Function (AF), (Radio) Access Network ((R)AN), Data Network (DN) and User Equipment (UE) [14].

The separation of the control and user planes guarantees that the resources of each plane can be scaled independently and that the user plane functions can be distributed, deployed and nearer to UE in order to shorten the latency requirements. 5G service based architecture model shortens the network path communication among the network functions and uniformly enables user services with different access systems, like fixed network access or WLAN, from the onset. Furthermore, the system architecture provides interworking with 4G, network capability exposure and numerous other functionalities.

The architecture proposed in this work is based on the 5G 3GPP standardized architecture. The design of this architecture considered two major requirements: scalability and latency. In order to satisfy both demands, in the proposal there are an SDN controllers hierarchy, the Area Controller (AC) in the bottom level and Global Controller (GC) in the top level. These controllers are used to bridge between the control and user plane, in this case specifically, between the SMF and the UPF. This sort of architecture guarantees scalability because we can define specific functions to each kind of controllers in order to define their role within the overall network. In this way, only traffic that is destined for other networks is moved to an GC. The communication between both kinds of controllers is achieved by the different applications that run on them.

Fig. 4 illustrates the 5G Control Plane architecture, in which the user plane entities, UPF, are implemented as OpenFlow switches and routers that can be also virtualized or remain as dedicated hardware. The concept of Agent is introduced in this plane taking into account the Fog Computing paradigm, as it seeks to place processing and storage resources closer to end users. Hence, these Agents are entry points to the networks.

A. SDN Controllers Hierarchy

As it was mentioned before, in the upper layer of the hierarchy are a group of GCs that are federated using DDS; they are in charge of managing and controlling the ACs, doing

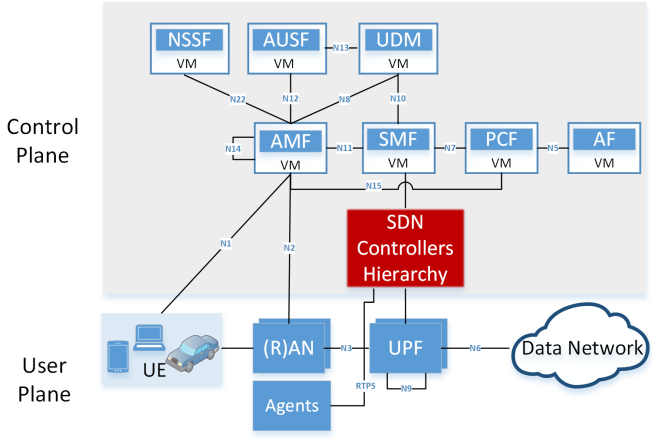

Fig. 4: Proposal of 5G Control Plane architecture

load balancing, keeping a global network view and so on. Although, the bottom layer is composed of ACs which are mainly responsible for the UPFs control and flows management.

Both kind of controllers use DDS to exchange network information. The GCs communicate with each other using DDS to keep a consistency network state, establish inter-domain flow routes, etc. Meanwhile, the ACs update their GCs by means of this application when occurs a change in their topology or in the state of their assigned nodes. Similarly, the GCs inform their ACs when there is a change in the global topology that affects the communication among nodes under the control of different ACs.

Not only do GCs and ACs use DDS to communicate among each other, but also the Agent use it to communicate with GCs and ACs. In this way, both GCs and ACs are notified about a service request and they can reply in a leisure time the service request. Through the use of DDS the latency requirement could be satisfied since both sort of controllers would be able to execute commands in a shorter time frame.

Furthermore, the use of DDS allows a better performance during recovery stages, because GCs will share its network information among each other. So, if any problem arises with a GC operation, its functions will be taken over by another GC.

The SDN Controllers hierarchy is depicted in Fig. 5. Similarly, it is shown the relation among the different modules that integrates the GC and the AC. These modules have different functions depending on the type of controller.

B. Global Controller Modules

Publisher/Subscriber Module: It sends or receives network information to or from others GCs or ACs. This module receives information from Agents related to different events that take place in the network, such as: link failures, broken controllers, service requests and so on. The Publisher/Subscriber runs the DDS App, which is a OMG's standard middleware for distributed real-time applications.

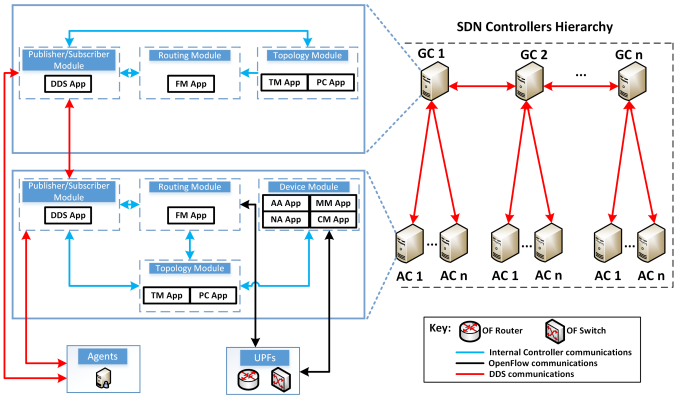

Fig. 5: Hierarchical architecture of federated SDN controllers.

Routing Module: It applies inter-area forwarding rules, meaning that, when an AC does not know which router to forward information to, this module will let it know. The Flow Management Application (FM App) runs on this module and is responsible for these functions.

Topology Module: It manages and storages information related not only to the AC connected to it, but also the GC. This module contains the Topology Management Application (TM App) and the Path Calculation Application (PC App). The first one manages a database with network information (nodes and links). Meanwhile, the second one runs an algorithm to calculate new routes based on the knowledge of physical infrastructure topology and utilization.

C. Area Controller Modules

Publisher/Subscriber Module: It executes identical functions to its homologue in the GC with the only difference that it does not send or receive information to or from other ACs.

Routing Module: The FM App runs on this module and applies intra-area forwarding rules taking into consideration network information.

Topology Module: The TM App and PC App run on this module. The first one manages a database with network information about switches and routers connected to the AC. The second one applies an algorithm to calculate new routes when there is any kind of failure or when switches or routers do not know how to forward a specific flow.

Device Module: It manages information about connected devices. Mobility Management Application (MM App), Connectivity Management Application (CM App), Authorization and Authentication Application (AA App) and Network Access Application (NA App) run on it to guarantee its functionality. MM App is in charge of handover procedure for wireless users. CM App and AA App are responsible for authentication procedure of new users in the network and for dealing with service requests. Last but not least, NA App attends the setup procedure between the user's device and the network. During this procedure service initialization parameters are established such as: Quality of Service (QoS), Bandwidth (BW) and so on.

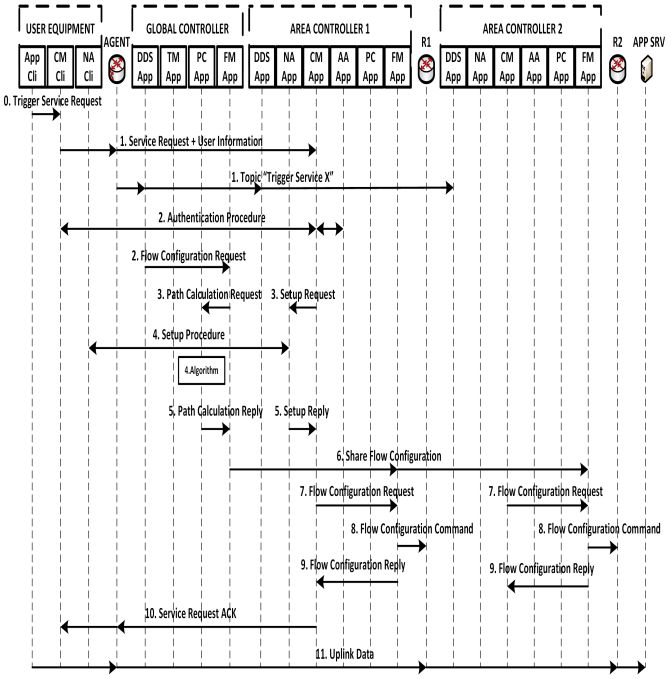

Fig. 6: Flow messages of Use Case 1.

V. Use Cases

In order to illustrate the functionality of each module of the Global and Area Controller in specific situations, we raise two use cases.

A. Use Case I: Service Requested to Different Area Controller

In this use case, it is explained how a user of an area controlled by AC1 can request a service running on other area controlled by AC2. This situation is described in Fig. 6.

Firstly, user's CM Client, triggered by an Application Client (App Client), sends a Service Request to CM App in AC1. This message passes through the Agent before reaching the CM App. The Agent recognizes a Service Request and collects the user information. At this moment, a Topic named "Trigger Service X" is generated by the Agent to its Subscribers who are the GC and the AC. In this way, these elements would quickly know that a UE of AC1 has requested a service of AC2, thus, the GC has to control those operations.

Some steps occur simultaneously in the GC and the AC1 after step 1. On one hand, in the GC, DDS App requests a flow configuration to FM App in order to define and implement rules to handle traffic between the UE and the service of AC2. In this phase, the flow configuration request is handled by FM and PC Apps (steps 3-5). On the other hand, in AC1, steps 3-5 involve to check the user identity, authorization and allocation of an IP address, if it is requested. It should be mentioned that the authentication procedure involve a communication between AC, GC and AUSF even though it does not appear in Fig. 6.

In step 6, the FM App of GC shares its flow configuration with FM App of AC1 and AC2, thus, both controllers will know how to handle the traffic between the UE and the App

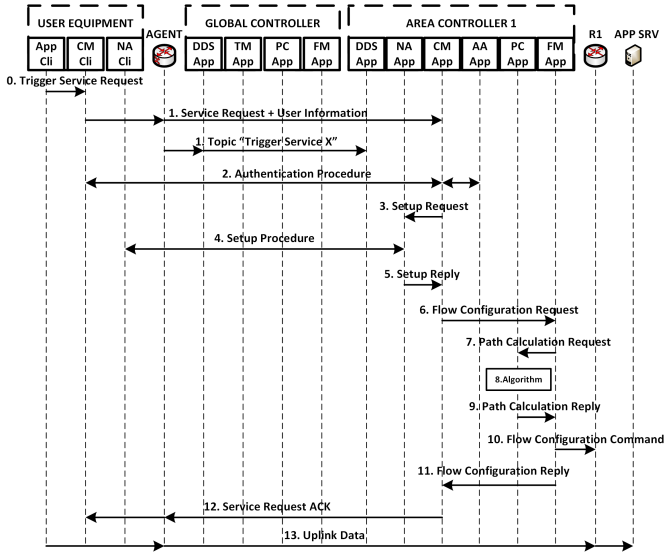

Fig. 7: Flow messages of Use Case 2.

Server. In steps 7-9, FM App of AC1 installs the forwarding rules in all network devices involved in the data traffic. When the forwarding path is completed, CM App acknowledges the service request to the UE in step 10. After that, the App Client starts sending data to the App Server.

B. Use Case II: Service Requested to User's Area Controller

In this case, a user from an area controlled by the AC1 requests a service running on the same area. This situation is depicted in Fig. 7.

The following steps are similar to those in the previous use case. The CM Client in the UE sends a Service Request to CM App in the AC1 during step 1. This message reaches the Agent that sends a Topic named "Trigger Service Y" to its subscribers. In this case, the GC does not perform any routing functionality, because the service and the UE are in the same area. Once the CM App receives the service request, steps 2-5 involve the authentication and setup procedure of the UE in the network. As it was mentioned in the previous use case, the authentication procedure involve a communication between AC, GC and AUSF even though it does not appear in Fig. 7.

At the completion of step 5, the CM App requests a flow configuration to FM App. This request is handled by the FM and the PC App in steps 7-9. All these steps do not need to run if the controller knows the forwarding path for this traffic. In this case, steps 10 runs immediately after step 6: FM App in the AC1 installs forwarding rules in all network devices involved in the data traffic. Then, once the User is acknowledged by CM App, the App Client starts sending data to the App Server.

VI. Conclusion

The network architecture for 5G has to include the necessary conditions to support many type of traffics and achieve a complete network and service convergence.

A hybrid hierarchy of SDN controllers was proposed in this article as a new architecture for the 5G Control Plane. In this architecture, a set of distributed and federated SDN controllers hierarchically superior control a group of SDN controllers in the bottom level. This design offers scalability, flexibility and programmability to the network, some of 5G requirements. Moreover, the latency requirement is solved by using DDS App in the proposal taking into account previous results in its use, as it sends large amount of data in a shorter time which it would allow to share information among GCs. By means of two use cases it is demonstrated how the proposal works in specific situations and the reliability of using DDS App in the proposal.

Acknowledgment

The authors would like to thank José Ángel Expósito-Arenas for their helpful ideas and constructive comments that greatly contributed to improving the overall quality of the article.

This work has been supported by the Ministerio de Economia y Competitividad of the Spanish Government under the projects TEC2016-76795-C6-1-R, TEC2016-76795-C6-4-R and AEI/FEDER, UE.

References

[1] I. Real-Time Innovations, "Rti connext dds core libraries user's manual 5.2.3." http://community.rti.com/static/documentation/connext-dds/5.2.3/doc/manuals/connext_dds/html_files/RTI_ConnextDDS_CoreLibraries_UsersManual/index.htm#UsersManual/title.htm%3FTocPath%3D_____1, 2016. [Online; accessed 20-October-2017].

[2] Y. Fu *et al.*, "A hybrid hierarchical control plane for flow-based large-scale software-defined networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 2, pp. 117–131, 2015.

[3] R. Trivisonno *et al.*, "Towards zero latency software defined 5g networks," in *Communication Workshop (ICCW), 2015 IEEE International Conference on*, pp. 2566–2571, IEEE, 2015.

[4] D. Camps *et al.*, "Analysis of state of the art on scalable control plane design and techniques for user mobility awareness. definition of 5g-xhaul control plane requirements," tech. rep., 5G-XHaul, 2016.

[5] V. Yazıcı *et al.*, "A new control plane for 5g network architecture with a case study on unified handoff, mobility, and routing management," *IEEE communications magazine*, vol. 52, no. 11, pp. 76–85, 2014.

[6] V.-G. Nguyen *et al.*, "Sdn/nfv-based mobile packet core network architectures: A survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1567–1602, 2017.

[7] A. Tootoonchian and Y. Ganjali, "Hyperflow: A distributed control plane for openflow," in *Proceedings of the 2010 internet network management conference on Research on enterprise networking*, pp. 3–3, 2010.

[8] T. Koponen *et al.*, "Onix: A distributed control platform for large-scale production networks.," in *OSDI*, vol. 10, pp. 1–6, 2010.

[9] Y. Fu *et al.*, "Orion: A hybrid hierarchical control plane of software-defined networking for large-scale networks," in *Network Protocols (ICNP), 2014 IEEE 22nd International Conference on*, pp. 569–576, IEEE, 2014.

[10] K. Phemius, M. Bouet, and J. Leguay, "Disco: Distributed multi-domain sdn controllers," in *Network Operations and Management Symposium (NOMS), 2014 IEEE*, pp. 1–4, IEEE, 2014.

[11] L. Bertaux, A. Hakiri, S. Medjiah, P. Berthou, and S. Abdellatif, "A dds/sdn based communication system for efficient support of dynamic distributed real-time applications," in *Distributed Simulation and Real Time Applications (DS-RT), 2014 IEEE/ACM 18th International Symposium on*, pp. 77–84, IEEE, 2014.

[12] A. Hakiri and A. Gokhale, "Data-centric publish/subscribe routing middleware for realizing proactive overlay software-defined networking," in *Proceedings of the 10th ACM International Conference on Distributed and Event-based Systems*, pp. 246–257, ACM, 2016.

[13] H.-Y. Choi, A. L. King, and I. Lee, "Making dds really real-time with openflow," in *Proceedings of the 13th International Conference on Embedded Software*, p. 4, ACM, 2016.

[14] 3GPP, "TS 23.501- System Architecture for the 5g System; Stage 2." http://www.3gpp.org/ftp/Specs/archive/23_series/23.501/23501-f00.zip. [Online; accessed 01-January-2017].